LOOPS: WHILE, FOR

Lab Instructor: Jean Lai
Computer can make repeated decisions on its own and performs an operation over and over again as long as a condition is true, such as following:

- Allow user three attempts for entering correct password.
- Calculate the letter grade for all students in the class.
- Calculate exponential powers \( n^n (n*n*...*n) \).

Two most commonly used loops:

1. **While** loop
2. **For** loop
**While construct**

```java
while (condition) {
    // do something
}
```

**Evaluation Steps**

1. **while** evaluates the condition
   - condition is *true* → continues to step 2.
   - condition is *false* → skips to step 4.
2. Executes the statements contained in the loop body. (statements contained within curly braces)
3. Repeats from Step 1.
4. Continues to the statements after the curly brace.
**LOOP CONDITION**

- Similar to conditions in **if** statement.

- Create the loop condition.
  1. Create a condition that can be evaluated to either **true** or **false**.
  2. Make sure there’s a way to exit the loop. That is, the condition may be updated to be false. (otherwise, it becomes an infinite loop, and program never terminates!)

- Utilize the similar comparison and logical operators.
What do we use to test the conditions?

- Use comparison operators to compare values.
- Resulting condition will return **true** or **false** value.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Meaning</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>==</td>
<td>Test for equality</td>
<td>x == y</td>
</tr>
<tr>
<td>!=</td>
<td>Test for inequality</td>
<td>x != y</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than</td>
<td>x &gt; y</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less than</td>
<td>x &lt; y</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater than or equal to</td>
<td>x &gt;= y</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or equal to</td>
<td>x &lt;= y</td>
</tr>
</tbody>
</table>
What if there’s more than one condition to test at once?

- Use logical operators – AND, OR, NOT
- Make sure there’s parentheses ( ) surrounding all the conditions!

<table>
<thead>
<tr>
<th>Logical Operator</th>
<th>Meaning</th>
<th>Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>&amp;&amp;</td>
<td>Test for BOTH conditions to be true</td>
<td>expr1 &amp;&amp; expr2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>!</td>
<td>Test for condition to be false</td>
<td>! expr</td>
</tr>
</tbody>
</table>
For now, we will utilize loops to perform repeated tasks such as

1. Printing the first 100 even numbers.
2. Perform the same calculation 10 times.
3. ...

Thus, we create a loop condition that utilizes counter to set the number of times needed to execute the statement block.

**Increment**

```c
int counter = 1;
counter = counter + 1;
counter += 1;
counter++;
++counter;
```

**Decrement**

```c
int counter = 1;
counter = counter - 1;
counter -= 1;
counter--;
--counter;
```
While construct

initialize loop variable

while (condition) {
    do something
    update loop variable
}

Example

int i = 1;
while (i < 10) {
    cout << i;
    i = i + 1;
}
Example

// Printing numbers from 1 to N.
int N;
cout << “Please input an integer.”;
cin >> N;

int i = 1;
while (i <= N) {
    cout << “i :” << i;
    i = i + 1;
}

Evaluation Steps

1. Get a number from keyboard and store in N.
2. Set i to 1.
3. while i is less than or equal to N:
   - i ≤ N → continue to step 4.
   - i > N → skip to step 7.
4. Print i.
5. Add 1 to i.
6. Repeat from Step 3.
7. Exit while loop.
FOR LOOP

- Similar to while loop.
- Controls program to execute repeated tasks.
- Construct 3 expressions at the same time – initialization, condition and update.
**For Construct**

```java
for (initialize; condition; update) {
    do something
}
```

**Evaluation Steps**

1. **Initialize** – initialize the loop variable value.
2. **Condition** - for evaluates the condition
   - condition is true $\rightarrow$ continues to statement block (step 3).
   - condition is false $\rightarrow$ skips statement block.
3. **Executes** the statement block.
4. **Update** – loop variable is updated.
5. Repeats from Step 2.
Example

// Printing numbers from 1 to N.
int N;
cout << "Please input an integer;";
cin >> N;

for (int i = 1; i <= N; i++) {
    cout << "i: " << i;
}

Evaluation Steps

1. Get a number from keyboard and store in N.
2. Set i to 1.
3. while i is less than or equal to N:
   - i ≤ N → continue to step 4.
   - i > N → skip to step 7.
4. Print i.
5. Add 1 to i.
6. Repeat from Step 3.
7. Exit while loop.